3-Heights™ PDF Optimization Shell

Version 4.5

User Manual

Contact: pdfsupport@pdf-tools.com

Owner: **PDF Tools AG**

Kasernenstrasse 1
8184 Bachenbülach
Switzerland

http://www.pdf-tools.com

Copyright © 2001-2015

Table of Contents

1 Introduction 5
1.1 Description 5
1.2 Functions 5
Features 6
Formats 7
Compliance 7
1.3 Operating Systems 7
2 Installation 8
2.1 Installing the 3-Heights™ PDF Optimization Shell 8
How to set the Environment Variable "Path" 8
3 License Management 9
3.1 Graphical License Manager Tool 9
List all installed license keys 9
Add and delete license keys 9
Display the properties of a license 10
Select between different license keys for a single product 10
3.2 Command Line License Manager Tool 10
List all installed license keys 10
Add and delete license keys 10
Select between different license keys for a single product 10
3.3 License Key Storage 10
Windows 11
Mac OS X 11
Unix / Linux 11
4 Getting Started and User's Manual 11
4.1 General Settings 11
4.2 Usage 12
4.3 Specify the Folder of the Output File 12
4.4 Processing All Files in a Folder 12
Windows Batch Sample 12
5 Optimization Process 15
5.1 Images 15
Relevant Factors for the File Size 15
Examples 16
Optimizing Raster Images 16
Mixed Raster Content (MRC) Optimization for Images 17
5.2 Fonts 19
5.3 Suggested Settings for the Web 19
5.4 Suggested Settings for Printing 20
6 Reference Manual 21
6.1 Compression Values 21
0 No Compression (Raw) 21
1 DCT (JPEG) Compression 21
2 Flate (ZIP) Compression 21

© PDF Tools AG – Premium PDF Technology

3 LZW (Lempel-Ziv-Welch) Compression 21
4 CCITT Fax Group 3 Compression 21
5 CCITT Fax Group 3 2D Compression 22
6 CCITT Fax Group 4 Compression 22
7 JBIG2 Compression 22
8 JPEG2000 Compression 22
9 MRC Optimization 23
10 Compression of the Source Image 23
6.2 Switches 24
-c Set the Color Conversion 24
-cff Compress Type1 fonts (convert to CFF) 25
-cms Set the color management engine 25
Resolution and Threshold Values per Image Type 25
-dr Set the Resolution in DPI 25
-dt Set the Threshold in DPI 26
-fb Set the Compression Types for Bi-tonal Images 26
-fc Set the Compression Types for Color and Grayscale Images 27
-ff Force Compression Conversion 28
-fi Set the Compression Types for Indexed (Paletted) Images 28
-fn Set File Name 28
-fv Set the Minimum PDF Version 28
-id Set Value in the Document Information Dictionary 29
-isc Enable Conservative Image Processing Strategy 29
-lf List Fonts 29
-li List Images 30
-lk Set License Key 31
-m Merge Embedded Font Programs 31
-ml Set Compression Type for MRC Layers 31
-mlr Set the Resolution in DPI for MRC Layers 31
-mlq Set the Image Quality for MRC Layers 32
-mm Set Compression Type for the MRC Mask 32
-mp Set Compression Type for MRC Cut-Out Pictures 32
-o Set the Owner Password 32
-oc Clip Images 32
-od Optimize Resources 32
-ol Linearize Only 33
-or Remove Redundant Objects 33
-ow Linearize the Output File 33
-p Set the Permission Flags 33
-pw Read an Encrypted PDF File 34
-q Set the Compression Quality 34
-rs Remove Embedded Standard Fonts 34
-s Subset Fonts 35
Strip the File 35
-u Set User Password 36
-v Verbose Mode 36
-xf Extract Fonts 36
-xi Extract Images 36
6.3 Return Codes 37

7 Troubleshooting 38

7.1 The Output File is Too Large 38

7.2 The Output File Is Larger Than the Input File.....39
7.3 The Selected Compression Type is Not Applied 39
7.4 The Output Document Is Not Encrypted.....39

© PDF Tools AG – Premium PDF Technology

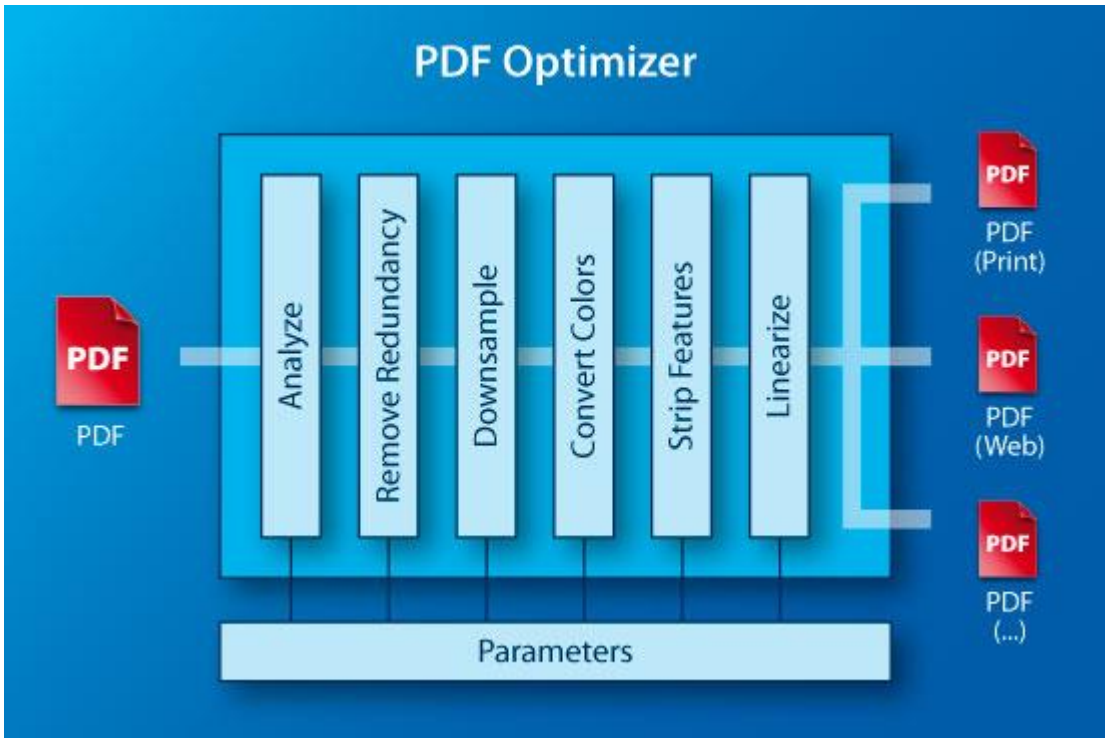
1 Introduction

1.1 Description

The 3-Heights™ PDF Optimization Shell optimizes PDF files to enable their use as high resolution files for printing or, with less resolution, for electronic document exchange or space-saving document archiving.

Many processes produce very large PDF files that are not suitable for electronic document exchange. Users are then tempted to convert the PDF documents into other formats, but this only makes the situation even worse. The correct approach, and the easiest, is to optimize large PDF documents.

This process optimizes fonts and images to the best possible size and quality. It also removes redundant document content and "linearizes" PDF documents to enable fast web display.



1.2 Functions

The use of the latest compression algorithms enables the tool to reduce the memory space requirements for images or lessen their resolution, remove redundant and alternative information, optimize fonts through summarization or subsetting, convert colors and linearize the PDF.

© PDF Tools AG – Premium PDF Technology

Features

- Optimization for Electronic Document Exchange, Web Publishing and Archiving
- Customized compression of bi-tonal, monochrome and color images
- Define image resolution in dots per inch
- Define threshold value for down-sampling
- Set the quality index of lossy compression
- Automatically select best compression type for images
- Perform mixed raster content (MRC) optimization for images
- Remove invisible parts of images
- Linearization (fast web display)
- Compile and subset fonts
- Read encrypted input files
- Encrypt and set access authorization for the output file
- Process memory-resident files
- Removal of:
    - Redundant objects
    - Obsolete objects stemming from previous changes to the file
    - Embedded standard fonts (e.g. Courier, Arial, Times)
    - Embedded, non-symbolic fonts
    - Unnecessary file information
    - Article threads
    - Alternative images
    - Metadata
    - Page piece information
    - Document structure tree including markup
    - Miniature page preview images
    - Spider (web capture) information
- Remove or clear form fields and annotations

Optimize for Printing:

- Color conversion (to RGB, CMYK or grayscale)
- Allow high print quality
- Set minimum PDF version of the output file

List and Extract Parameters:

- Fonts and their properties
- Images and their properties

© PDF Tools AG – Premium PDF Technology

- Error Code
- Number of pages

Formats

Input Formats:

- PDF 1.x (e.g. PDF 1.4, PDF 1.5.)

Target Formats:

- PDF 1.x (e.g. PDF 1.4, PDF 1.5)

Compliance

Standards: ISO 32000 (PDF 1.7)

1.3 Operating Systems

- Windows XP, Vista, 7, 8, 8.1 - 32 and 64 bit
- Windows Server 2003, 2008, 2008 R2, 2012, 2012 R2 - 32 and 64 bit
- HP-UX 11 and later PA-RISC2.0 32 bit or HP-UX 11i and later ia64 (Itanium) 64 bit
- IBM AIX 5.1 and later (64 bit)
- Linux (32 and 64 bit)
- Mac OS X 10.4 and later (32 and 64 bit)
- Sun Solaris 2.8 and later, SPARC and Intel
- FreeBSD 4.7 and later 32 bit or FreeBSD 9.3 and later 64 bit (on request)

© PDF Tools AG – Premium PDF Technology

2 Installation

2.1 Installing the 3-Heights™ PDF Optimization Shell

The retail version of the 3-Heights™ PDF Optimization Shell comes as a ZIP archive containing various files including runtime binary executable code, documentation and license terms.

1. Download the ZIP archive of the product from your download account at www.pdf-tools.com.
2. Open the ZIP archive.
3. Check the appropriate option to preserve file paths (folder names) and unzip the archive to a local folder (e.g. *C:\program files\pdf-tools*).
4. The unzip process now creates the following subdirectories:
   - *Bin*: Contains the runtime executable binary code
   - *Doc*: Contains documentation files
5. To start the 3-Heights™ PDF Optimizer Tool from a shell, the directory needs to be included in the "Path" environment variable.

How to set the Environment Variable "Path"

To set the environment variable "Path" on Windows 2000, go to **Start** -> **Settings** -> **Control Panel** -> **System** -> **Advanced** -> **Environment Variables**

Windows XP, go to **Start** -> **Control Panel** (classic view) -> **System** -> **Advanced** -> **Environment Variables**.

Select "Path" and **Edit**, then add the directory where *pdfoptimize.exe* is located to the "Path". If the environment variable "Path" does not exist, create it.

© PDF Tools AG – Premium PDF Technology

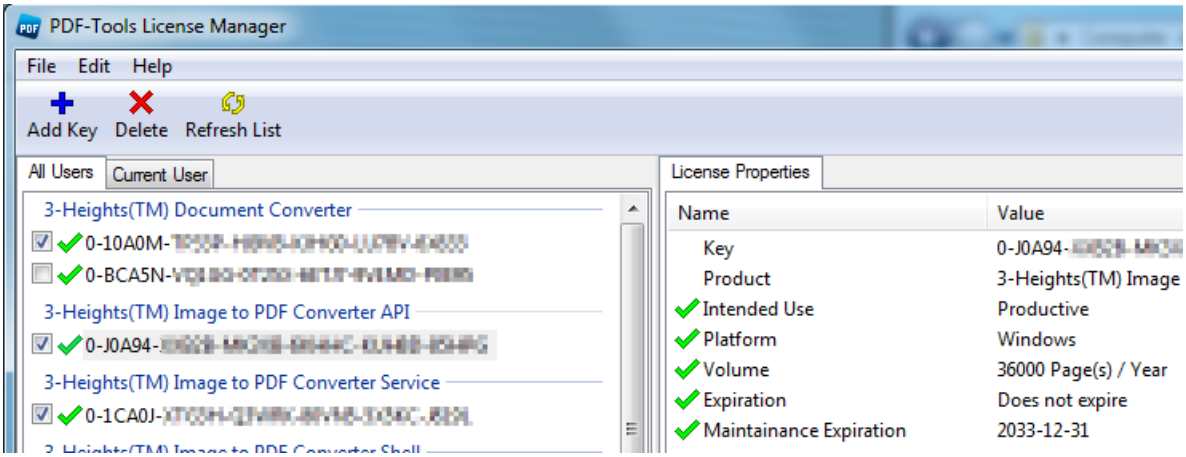
3 License Management

There are three possibilities to pass the license key to the application:

1. The license key is installed using the GUI tool (Graphical user interface). This is the easiest way if the licenses are managed manually. It is only available on Windows.
2. The license key is installed using the shell tool. This is the preferred solution for all non-Windows systems and for automated license management.
3. The license key is passed to the application at runtime via the command line switch `-lk` property. This is the preferred solution for OEM scenarios.

3.1 Graphical License Manager Tool

The GUI tool *LicenseManager.exe* is located in the *bin* directory of the product kit.



List all installed license keys

The license manager always shows a list of all installed license keys on the left pane of the window. This includes licenses of other PDF Tools products.

The user can choose between:

- Licenses available for all users. Administrator rights are needed for modifications.
- Licenses available for the current user only.

Add and delete license keys

License keys can be added or deleted with the "Add Key" and "Delete" buttons in the toolbar.

- The "Add key" button installs the license key into the currently selected list.
- The "Delete" button deletes the currently selected license keys.

Display the properties of a license

If a license is selected in the license list, its properties are displayed in the right pane of the window.

Select between different license keys for a single product

More than one license key can be installed for a specific product. The checkbox on the left side in the license list marks the currently active license key.

3.2 Command Line License Manager Tool

The command line license manager tool *licmgr* is available in the *bin* directory for all platforms except Windows.

A complete description of all commands and options can be obtained by running the program without parameters:

```
licmgr
```

List all installed license keys

```
licmgr list
```

The currently active license for a specific product is marked with a star '*' on the left side.

Add and delete license keys

Install new license key

```
licmgr store X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Delete old license key

```
licmgr delete X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Both commands have the optional argument `-s` that defines the scope of the action:

- `g`: For all users
- `u`: Current user

Select between different license keys for a single product

```
licmgr select X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

3.3 License Key Storage

Depending on the platform the license management system uses different stores for the license keys.

Windows

The license keys are stored in the registry:

- HKLM\Software\PDF Tools AG (for all users)
- HKCU\Software\PDF Tools AG (for the current user)

Mac OS X

The license keys are stored in the file system:

- /Library/Application Support/PDF Tools AG (for all users)
- ~/Library/Application Support/PDF Tools AG (for the current user)

Unix / Linux

The license keys are stored in the file system:

- /etc/opt/pdf-tools (for all users)
- ~/.pdf-tools (for the current user)

Note: The user, group and permissions of those directories are set explicitly by the license manager tool.

It may be necessary to change permissions to make the licenses readable for all users. Example:

```
chmod -R go+rx /etc/opt/pdf-tools
```

4 Getting Started and User's Manual

The simplest command requires two parameters: The names of the PDF input and output files.

```
pdfoptimize input.pdf output.pdf
```

This command will generate an new PDF file with optimized images based on the default compression values for bi-tonal, monochrome (grey scale) and color images (see the "Reference Manual" chapter for default values).

4.1 General Settings

Pass a license key to the application at runtime instead of installing it on the system.

```
pdfoptimize -lk X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX input.pdf
output.pdf
```

This is only required in an OEM scenario.

4.2 Usage

By typing **pdfoptimize** without parameters, the usage, the version and a list of available options is returned.

4.3 Specify the Folder of the Output File

The output folder can simply be added in front of the output file name

```
pdfoptimize input.pdf myfolder\output.pdf
```

4.4 Processing All Files in a Folder

If you would like to process all files in a directory, it is required to use a variable to name the output files. Here is an example using the FOR command of the CMD shell (see also **for /?** for additional help) and the variable %i. It optimizes all *.pdf files in the current directory and saves them with the appendix "_opt", in the same folder:

```
for %i in (*.pdf) do pdfoptimize –v -or %i %~ni_opt.pdf
```

If you would like to keep the file name, the output documents need to be created into another folder. The input file cannot be overwritten directly due to the fact that the optimization process reads from the input file, while it already writes to the output file.

```
for %i in (C:\in\*.pdf) do pdfoptimize –or %i C:\out\%~ni.pdf
```

When using variables in a batch file (.bat), variables have 2 leading % instead of just 1 like on the command line.

Windows Batch Sample

In a situation where all files in a directory need to be processed and the optimized file should have the same name as the original document, i.e. overwrite it, the following approach can be used.

- Make sure you really want this, the original file is lost in this process!
- Create the output files, either with a different name or in a different directory.
- Ensure the output files are created correctly. This can be done by verifying the return code (must be 0), or verify the document was created at all and is not empty.
- Delete the original file.
- Rename or copy back the new file to replace the original file.

The following sample does the steps described above. This sample does not ensure to always yield a correct result. Errors in the optimization or an abort of the process can still lead to loss of data. It is suggested to keep a backup of the original files.

```
@ECHO off
rem  ***************************************************************************
rem  *  This batch files optimizes all PDF files in the current directory  *
```

```
rem  *  ------------------------------------------------------------------  *
rem  *  The steps are as following:                                         *
rem  *                                                                      *
rem  *    1. Optimize all files in a folder. The optimized output files     *
rem  *       have the temporary extension .tmp.                             *
rem  *                                                                      *
rem  *    2. If the return code of the pdfoptimize is 0, and an output is   *
rem  *       created, the optimization process is considered successful.    *
rem  *                                                                      *
rem  *    3. If successful, the original input file is deleted and the      *
rem  *       .tmp file is renamed to .pdf.                                  *
rem  *                                                                      *
rem  *       If the process was not successful, the .tmp file is deleted    *
rem  *       and the original file is left as is.                           *
rem  ************************************************************************

IF EXIST *.tmp DEL /F /Q *.tmp
FOR %%i in (*.pdf) DO (
    SET name=%%~ni
    CALL :_Optimize
)
GOTO :EOF
rem  ************************************************************************
:_Optimize
pdfoptimize -or "%name%.pdf" "%name%.tmp"
IF NOT %ERRORLEVEL%==0 (
    @ECHO ** Optimization process failed for %name%.pdf [error code %ERRORLEVEL%].
    IF EXIST "%name%.tmp" DEL /F /Q "%name%.tmp"
) ELSE (
    IF EXIST "%name%.tmp" (
        IF EXIST "%name%.pdf" (
            DEL /F /Q "%name%.pdf"
            IF NOT EXIST "%name%.pdf" (
                RENAME "%name%.tmp" "%name%.pdf"
                @ECHO ** Optimization process successful for %name%.pdf.
            ) ELSE (
                DEL /F /Q "%name%.tmp"
                @ECHO ** Optimization process failed for %name%.pdf [file locked].
            )
        )
```

```
    ) ELSE (
        @ECHO ** Optimization process failed: %name%.pdf [error code %ERRORLEVEL%].
    )
)
GOTO :EOF
```

In order to optimize all files in all sub-folders, it's easiest to create a batch file that runs through all sub-folders and executes the batch file above.

So, create a batch file called *run.bat* and copy the upper code in it.

Then create another batch file called for example *runsub.bat* and add the code below:

```
@ECHO OFF
FOR %%r IN (.\) DO SET rootfolder=%%~pr
FOR /R %%s IN (.) DO (
  CD %%s
  CALL %rootfolder%run.bat
)
CD %rootfolder%
SET rootfolder=
```

Now copy the two batch files to the root folder (i.e. the folder from which every PDF file in every sub folder should be processed) and run the batch *runsub.bat*.

5 Optimization Process

The main intent of the 3-Heights™ PDF Optimization Tool is to reduce the file size of a PDF document. The file size heavily depends on the document's content and on how the PDF is constructed internally. Often embedded font programs, embedded color profiles, and images have the highest impact.

Thus, the 3-Heights™ PDF Optimization Tool can be used to optimize a PDF for a specific field of application (e.g. Internet, Printing, etc.). For that purpose it offers various options to optimize and remove parts of the PDF.

5.1 Images

Relevant Factors for the File Size

The size of an image is basically determined by four factors:

1) The pixel mass: The total amount of pixels the image has. An image with a size of 600 by 800 pixels has 480000 pixels total.
2) The color depth: How many bits are required to describe 1 pixel? The table below gives the answer for different types of images. For example, an RGB image with 600 by 800 pixels requires therefore 600 x 800 x 3 bytes = 1.44 Mbytes in uncompressed format.

Color Space	***Description***	***Bits/pixel***
Bi-tonal	Black and white	1
Indexed	Colors are stored in an index table which usually holds 2 to 256 entries, e.g. GIF.	2-8
Grayscale	Monochrome	8
Color RGB	Color using Red, Green, Blue	24
Color CMYK	Color using Cyan, Magenta, Yellow, Key (=black)	32

3) The compression type: A compression algorithm can compress data (such as an image) to reduce its file size. Such an algorithm belongs to either of the following two classes:
    a. Lossless: The original image can be restored exactly.
    b. Lossy: The compression modifies the pixels. The original image cannot be restored from the compressed version. This is typically applied to photographic images where the human eye cannot distinguish whether the image was modified. The most common lossy compression is JPEG. The benefit of lossy compression is the higher compression ratio.

See also chapter "Supported image compression types".

4) The content of the image: The simpler the image, the better it compresses. For most compression algorithms a simple image (e.g. completely white) compresses much better than a complex image (e.g. a photo).

Examples

CCITT Fax compression was designed to compress black text written on a white background. The algorithm was optimized under the assumption that a page contains more white pixels than black pixels. Therefore a bi-tonal image with a lot of black does generally not compress as well as in image with more white even if they have the same pixel mass.

JBIG2 compression searches for patterns, and uses them multiple times. For example in a scanned text document the same few dozen of characters are used over and over again. The algorithm is optimized to save frequent patterns more efficiently than rare ones.

Optimizing Raster Images

The 3-Heights™ PDF Optimization Tool offers the following possibilities to optimize images:

1) The pixel mass can be reduced. (It cannot be increased.) This is done by clipping (cropping) the image size to its visible extent and/or by reducing the image resolution.

   The resolution defines how many pixels there are in given length of the image. The most common unit for resolution is dpi: Dots per inch. If an image has a resolution of 200 dpi, it means when displayed at 100% zoom, there are 200 pixels for 1 inch of image. The higher the resolution is, the "sharper" the image. A monitor has usually a resolution of at least 96 dpi, a laser printer of at least 600 dpi. When the file size matters, a common resolution for color and grayscale images in PDF is 150 dpi (usually higher for bi-tonal).

   The process of changing the amount of pixels an image has, is called re-sampling, or down-sampling when the result has less pixels than the original image.

   In the 3-Heights™ PDF Optimization Tool down-sampling is applied by setting a target resolution and a threshold resolution. The default values are 150 dpi for the target resolution and 225 dpi for the threshold resolution. This means every image that has a resolution of 225 dpi or higher is potentially down-sampled to 150 dpi. Of course, the threshold resolution can be set equal to the target resolution. However there are many cases where down-sampling by just a little bit has disadvantages. In particular, lossy images (e.g. JPEG compression) loose visual quality every time they are newly compressed. On top of that the compressed output can be larger than the input because artifacts introduced by the previous compression(s) are now considered as part of the image which needs to be compressed and lead to a worse compression even when the resolution is reduced. Per default, the 3-Heights™ PDF Optimization Tool will, however, prevent such unnecessary re-sampling.

2) The color depth can be modified for color images. The color depth can be left unchanged, set to Grayscale (8 bit), RGB (24 bit) or CMYK (32 bit). It cannot be

changed to black and white (1 bit). Note that in certain circumstances, the color depth of the image is not converted, e.g., if the resulting file size increases or if the image is pre-blended with a matte color.

3) The compression can be setup independently for the following three image compression types:

Type	*Description*
Bi-tonal	Black and white images.
Indexed	Images with an indexed (also known as "paletted") color space.
Continuous	Color (RGB and CMYK) images and grayscale images.

Bi-tonal images usually contain text or black and white graphics, indexed images usually contain color graphics such as logos, while continuous images usually contain photographs.

For each of the above image types, several compression algorithms can be set. The 3-Heights™ PDF Optimization Tool tries all the given compression algorithms and takes the one that yields the smallest file size. Note that the more compression algorithms are set, the longer the process of optimizing images will take.

Furthermore, a more conservative image processing strategy can be enabled. This strategy prevents all the compression trials if the image has neither been clipped nor down-sampled nor undergone a color-conversion. Hence, if the image has not been altered, then the original image from the input document is taken.

4) The content of the image cannot be changed directly. However changing the resolution or applying a lossy compression algorithm modifies the content of the image.

Note: Unless forcing of re-compression is enabled, the 3-Heights™ PDF Optimization Tool never increases the file size of an image because it chooses the smallest among all tried compression algorithms and the original image in the input file. This means the 3-Heights™ PDF Optimization Tool cannot be used to "un-compress" embedded images.

Mixed Raster Content (MRC) Optimization for Images

Some raster images – typically scanned documents – consist mainly of text, possibly in several colors and interspersed with some pictures. Such images are difficult to compress with one single compression type because of the diverse or even conflicting features of different parts of the image.

MRC optimization is a way of breaking such images down into parts, such that each part is well suited for one type of a compression algorithm. With this approach, the resulting file size often can be reduced without significantly reducing the visual quality of the document.

Note: MRC optimization can only be enabled for continuous images, i.e. not for bi-tonal images and images with an indexed color space.

MRC optimization may yield unexpected results, e.g. because the input image is not suitable for MRC. As another example, images in the original PDF may be stored as

small slices, and MRC optimization fails because the 3-Heights™ PDF Optimization Tool has no option to concatenate such image slices.

A PDF that contains MRC-optimized images is not suited for optical character recognition (OCR) and image extraction.

In the 3-Heights™ PDF Optimization Tool, MRC optimization works in three phases as explained below.

Phase 1: Cutting out Pictures

In this phase, the input image is analyzed and rectangular areas containing photographic features are detected. Each detected region is cut out and placed as a separate image in the resulting PDF.

Depending on the input image it is possible that this phase decides that the whole input image consists of one photographic region covering the whole image. In this case, the second phase (Phase 2: Separation into Layers) is omitted.

On the other hand, it is possible, that actual photographic regions present in the input image are not recognized correctly. This can happen for example if a photographic region contains parts with uniform color.

For the cut-out images, a compression type can be set.

Note: The resulting cut pictures are neither down-sampled nor color-converted.

This first phase is optional and can be switched off.

Phase 2: Separation into Layers

For this second phase the image is not supposed to contain any photographic features. Instead, the image is assumed to consist of text and graphic, potentially with varying color.

Now, the whole image is separated into two layers, a foreground and a background layer. Additionally, a mask is created, which can be thought of as a bi-tonal image that is not displayed directly but tells for each pixel whether to show the foreground layer or the background layer.

Example: Let the image consist of a yellow background with black paragraph text and a title text in red. Then the resulting background layer contains the yellow color only. The foreground layer contains the black text color where the paragraph text is located and the red text color where the title is located. In the mask, pixels for which the foreground layer should be displayed are set to 1, the others are set to 0. I.e. the mask contains 1's where the black and the red text is and 0's everywhere else.

In the resulting PDF the foreground layer, the background layer and the mask are stored as three images and thus are allowed to have different resolution and different compression types. Since all the detailed features have been moved to the mask, it makes sense to down-sample the foreground and background layers and use a low image quality. The mask on the other hand is usually stored with a lossless compression type optimized for text.

Phase 3: Reconstruction

In this phase the results of phase 1 (the cut-out images) and phase 2 (the layers and the mask) are used to synthesize the desired result. If in phase 1, a single

© PDF Tools AG – Premium PDF Technology

photographic region covering the entire image is detected, then the original image is used and the reconstruction is finished. Otherwise, the reconstruction first places the background layer, followed by the foreground layer with the mask. Finally if any cut-images are found they are placed at their respective locations on top of the foreground layer.

5.2 Fonts

Every text in a PDF document is written with a font. This font can either be embedded or not embedded in the resources of the PDF. Embedded means a font program is embedded that describes how glyphs are drawn. If a font is not embedded the application rendering the PDF (e.g. 3-Heights™ PDF Viewer or Adobe Acrobat) have to select a replacement font. Therefore the visual appearance of text written with an embedded font is determinable, whereas it is not when the font is not embedded.

A font program can be quite large. An embedded font which contains all WinAnsi characters has a size of about 20-100 kilobytes, if it contains a large Unicode range (e.g. Asian Characters) it can be several megabytes, whereas a non embedded font requires much less.

This leads to the following ways to optimize fonts:

1) Remove the embedded Font: Removing embedded fonts can reduce the file size of a document, particularly when the document contains many fonts. Removing fonts is best applied to (PDF-) standard fonts, such as Arial, Courier, Courier New, Helvetica, Times, Times New Roman. Removing fonts should not be applied to barcode fonts or fancy types.

   Note: PDF/A requires fonts to be embedded.

2) Subset Fonts: Only keep the information in the font program that is required to render the characters that are actually used in text in this document. All unused characters are removed.

3) Merge Fonts: A document can have the same font, or a subset of it, embedded multiple times. This commonly occurs when multiple input document, are merged into one large output document. The 3-Heights™ Optimization Tool can merge these fonts into one font (if they can be merged).

5.3 Suggested Settings for the Web

When optimizing PDF files for the web, the main goal is to reduce the file size without losing too much visual quality. Additionally files should be linearized, which allows for viewing random pages without download the entire file.

Suggested settings:

```
-c 1 -fb 6,7 -fc 1,8 -oc -od -or -ow -s -m -rs
```

Additionally, optional information can be stripped to further minimize the file size:

```
-sa -si -sm -ss -ss -st -sw
```

If encrypting:

```
-o <ownerpassword> -p pf
```

5.4 Suggested Settings for Printing

Suggested settings:

```
-c 2 -dt -1 -od -or -s -m
```

Optionally, information can be stripped to further minimize the file size:

```
-sa -sf -sm -ss -ss -st -sw
```

If encrypting:

```
-o <ownerpassword> -p pd
```

© PDF Tools AG – Premium PDF Technology

6 Reference Manual

6.1 Compression Values

0 No Compression (Raw)

The raw format results in an uncompressed image. Applying raw does not uncompress already compressed images.

Compression	None
Color depth	any

1 DCT (JPEG) Compression

The DCT (Discrete Cosine Transformation) is commonly used for image processing, especially for lossy data compression.

Compression	High, Lossy
Color depth	8, 24
Application area	Color images

2 Flate (ZIP) Compression

Flate is a lossless data compression algorithm that uses a combination of the LZ77 algorithm and Huffman coding.

Compression	High, Lossless
Color depth	8, 24
Application area	Images

3 LZW (Lempel-Ziv-Welch) Compression

LZW (Lempel-Ziv-Welch) is an implementation of a lossless data compression algorithm created by Abraham Lempel and Jacob Ziv. It was published by Terry Welch in 1984 as an improved version of the LZ78 dictionary coding algorithm developed by Lempel and Ziv. There are certain countries where this algorithm is still protected by a copyright.

LZW compression is prohibited in PDF/A-1.

Compression	High, Lossless
Color depth	2-8
Application area	Grey-scale images, artificial images

4 CCITT Fax Group 3 Compression

1-dimensional version of the CCITT Group 3 Huffman encoding algorithm.

© PDF Tools AG – Premium PDF Technology

Compression	Low, Lossless
Color depth	1
Application area	Line-art image, bi-tonal, faxes

5 CCITT Fax Group 3 2D Compression

2-dimensional version of the CCITT Group 3 Huffman encoding algorithm. It provides a higher compressed ratio than CCITT Group 3.

Compression	Medium, Lossless
Color depth	1
Application area	Line-art image, bi-tonal, faxes

6 CCITT Fax Group 4 Compression

An advanced version of a bi-tonal algorithm based on the CCITT Fax Group 3 2D compression. This compression provides generally the best compression of all CCITT Fax compressions.

Compression	Medium, Lossless
Color depth	1
Application area	Line-art image, bi-tonal, faxes

7 JBIG2 Compression

JBIG2 is an image compression standard for bi-level images, developed by the Joint Bi-level Image Experts Group. It is suitable for both lossless and lossy compression. It provides the best compression of all bi-tonal compression algorithms. Depending on the image the compression ratio is up to twice as good as CCITT Group 4, however due to its complexity, it also takes more time to compress and uncompress.

JBIG2 compression requires PDF version 1.4 or later.

Compression	High, Lossless (Q=100)/Lossy
Color depth	1
Application area	Line-art image, bi-tonal

8 JPEG2000 Compression

JPEG 2000 is a wavelet-based image compression standard. It was created by the Joint Photographic Experts Group committee with the intention of superseding their original discrete cosine transform-based JPEG standard.

JPEG2000 compression requires PDF version 1.5 or later, it is prohibited in PDF/A-1.

Compression	High, Lossless (Q=100)/Lossy
Color depth	8, 24
Application area	Images

© PDF Tools AG – Premium PDF Technology

9 MRC Optimization

In contrast to the values 0-7, this is not a single compression format. Instead, this enables MRC optimization on color and monochrome images. (See Mixed Raster Content (MRC) Optimization for Images.)

Application area	Scanned documents

10 Compression of the Source Image

In contrast to the values 0-7, this is not a single compression format. Instead, this tells 3-Heights™ PDF Optimization Shell to use the same compression as the original input image.

© PDF Tools AG – Premium PDF Technology

6.2 Switches

Switches are options that are provided with the command to define how the document should be optimized. Switches are listed in alphabetical order in this chapter.

Switches can occur in two forms: As stand-alone option, such as `-od` (optimize resources) or they may require a parameter, such as `-q 80` (set compression quality index to 80).

The last two parameters of the command line should always be the input and the output-file. (There is no output-file required when using any of the listing-options.)

Switches are parsed from left to right, the last set value is applied.

Example: *The following command sets the resolution for re-sampling of all raster image types (color, monochrome, bi-tonal) to 100, then it resets the monochrome resolution explicitly to 120.*

```
pdfoptimize -dr 100 -dmr 120 input.pdf output.pdf
```

If in the above command the setting –dmr 120 was set before –dr 100, it would not have any influence, since –dr 100 applies to all compressions and therefore would overwrite the previous setting.

-c Set the Color Conversion

This switch allows for converting raster images from one color space into another. E.g. it allows for converting all RGB images to CMYK images.

This switch does not have any impact on objects other than raster images that use color spaces, such as vector graphics or text. Color key masked images are not color converted. Pre-blended images can be converted from RGB to Grayscale, if the force conversion feature is set.

Use the switch `-c` followed by one of the parameters in the table listed below:

Table: Color Conversion

Parameter	*Conversion*	*Color values*
0 default	Don't convert colors	
1	Convert to ICE sRGB colors	red, green, blue
2	Convert to CYMK color (using profiles)	cyan, yellow, magenta, key
3	Convert color images to grey scale	grey

Example: *To convert all embedded color images that use the RGB color space to images of the CMYK color space, use the following command:*

```
pdfoptimize -c 2 input.pdf output.pdf
```

-cff Compress Type1 fonts (convert to CFF)

Convert embedded Type1 (PostScript) fonts to Type1C (Compact Font Format). This reduces the file size.

-cms Set the color management engine

Set the Color Management System (CMS) Engine used for color conversion (option –c). Supported engines are:

- *"None":* No CMS is applied. This results in the maximum possible contrast.
- *"Neugebauer":* The Neugebauer algorithm efficiently converts CMYK to RGB. It does not need any color profiles. The results however look similar to conversion using color profiles.
- *"MSICM":* The Microsoft ICM Engine.
- *"lcms" (default):* Use ICC color profiles (default profiles are used for all unmanaged device color spaces).
- *FileName*: Providing a file name, a configurable version of the Neugebauer algorithm is applied. The coefficients can be defined in the text file. The Neugebauer algorithm mixes the colors based on the amount of color and the corresponding weighted coefficient. Altering the values for a pure color specifically changes the result for this pure color. The color transition remains smoothly.

Resolution and Threshold Values per Image Type

The target resolution values can be set individually for different types of images using the following switches followed by a numerical parameter (default: 150):

-dbr Target resolution for bi-tonal (black and white) images

-dcr Target resolution for color images

-dmr Target resolution for monochrome (grey scale) images

The threshold values can be set with these switches followed by a numerical parameter (default 225):

-dbt Threshold resolution for bi-tonal images

-dct Threshold resolution for color images

-dmt Threshold resolution for monochrome images

For examples refer to switches **-dr** and **-dt**.

-dr Set the Resolution in DPI

Set the target resolution after re-sampling in dots per inch (dpi). Only those images with a resolution value higher than the threshold value, which is set with option **-dt**, will be processed. The default target resolution is 150 dpi. Pre-blended images, images with a color key mask, masks, and soft mask images are not re-sampled.

© PDF Tools AG – Premium PDF Technology

Example: *In order to down-sample all raster images with a resolution greater than 150 dpi to 75 dpi, apply the following:*

```
pdfoptimize –dt 150 –dr 75 input.pdf output.pdf
```

-dt Set the Threshold in DPI

This switch defines the minimum resolution an image must have to be optimized. The threshold value for re-sampling raster images is used in conjunction with the switch **–dr**, which sets the actual target resolution for those re-sampled images.

The threshold resolution must be equal or higher than the target resolution. If the value is set to –1, re-sampling is turned off. The default threshold resolution is 225 dpi.

Example*: Down-sample all raster images with an original resolution higher or equal to 150 dpi to a new resolution of 75 dpi:*

```
Pdfoptimize –dt 150 –dr 75 input.pdf output.pdf
```

Example: *To disable re-sampling, set the threshold value, set it to –1.*

```
pdfoptimize –dt –1 input.pdf output.pdf
```

If the size (in terms of bytes) of the re-sampled image is larger than its original size, the original image is kept instead.

-fb Set the Compression Types for Bi-tonal Images

This option affects only bi-tonal (black and white) images. The switch **-fb** is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Table: Bi-tonal Compression Types

Parameter	*Compression Filter*
0	RAW data
2	Flate (ZIP) compression
3	Lempel-Ziv-Welch (LZW) compression
4	CCITT Fax Group 3 compression
5	CCITT Fax Group 3 2D compression
6 default	CCITT Fax Group 4 compression
7	JBIG2 compression
10	Take the compression type from the original image in the input PDF

Example: *To let the 3-Heights™ PDF Optimization Shell try CCITT Group 3 compression, JBIG2 compression and the compression that is used in the source image of the original file use the following command:*

```
pdfoptimize -fb 3,7,10 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimization Shell go through all bi-tonal images and processes each image individually as follows. All the given compression algorithms are executed. If the input image has a compression different from CCITT Fax Group 3 and JBIG2, then the compression of the input image is also executed. As a result, several candidate versions are obtained. Now a choice is made among all these versions and the original image based on the size in bytes. The smallest candidate is chosen and used in the output document.

-fc Set the Compression Types for Color and Grayscale Images

This option affects normal color images (RGB and CMYK) as well as grayscale (monochrome) images. The switch `-fc` is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Table: Color / Monochrome Compression Types

Parameter		*Compression Filter*
0		RAW data
1	default	DCT (JPEG) compression
2		Flate (ZIP) compression
8		JPEG2000 compression
9		Perform MRC optimization (See Mixed Raster Content (MRC) Optimization for Images)
10		Take the compression type from the original image in the input PDF

Example*: : *To let the 3-Heights™ PDF Optimization Shell try JPEG compression, JPEG2000 compression and the compression that is used in the source image of the original file use the following command:

```
pdfoptimize -fc 1,8,10 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimization Shell go through all color and grayscale images and processes each image individually as follows. All the given compression algorithms are executed. If the input image has a compression different from JPEG and JPEG2000, then the compression of the input image is also executed. As a result, several candidate versions are obtained. Now a choice is made among all these versions and the original image based on the size in bytes. The smallest candidate is chosen and used in the output document.

Example*: *The following command performs MRC optimization for color and grayscale images:

```
pdfoptimize -fc 9 input.pdf output.pdf
```

If the result takes more memory than the original image in the input document, then the original image is used. See also Mixed Raster Content (MRC) Optimization for Images.

-ff Force Compression Conversion

If set, images are always recompressed. If not set (default), images are only recompressed if the resulting image is smaller than the original, i.e. occupies less bytes to store in the file.

-fi Set the Compression Types for Indexed (Paletted) Images

This affects only images with an indexed color space. This type of color space is sometimes used for color graphics and logos. The switch **-fc** is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Table: Indexed Compression Types

Parameter		*Compression Filter*
0		RAW data
2	default	Flate (ZIP) compression
3		Lempel-Ziv-Welch (LZW) compression
10		Take the compression type from the original image in the input PDF

***Example*: :** *To let the 3-Heights™ PDF Optimization Shell try Flate compression and LZW compression use the following command:*

```
pdfoptimize -fi 2,3 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimization Shell go through all images with indexed color space and processes each image individually as follows. All the given compression algorithms are executed. As a result, two candidate versions are obtained. Now a choice is made among these two versions and the original image based on the size in bytes. The smallest candidate is chosen and used in the output document.

-fn Set File Name

The intension of this switch is to provide support for file names that start with a dash character and would therefore cause a parameter error.

The parameter after the switch **-fn** is a file name. It can optionally also be used for file names not starting with a dash character.

Example:

```
pdfoptimize -fn -input.pdf output.pdf
```

-fv Set the Minimum PDF Version

This option allows for setting the minimum PDF version of the created PDF output file. Supported values are 1.1 to 1.7. (PDF 1.4 corresponds to Acrobat 5, PDF 1.5 to Acrobat 6, etc.) There are three parameters that influence the version of the PDF output file:

- The value set as parameter of the switch **-fv**
- The PDF version of the input file
- Other settings in the optimization (i.e. JBIG2 requires PDF 1.4, JPEG2000 requires PDF 1.5)

The maximum of the three values above sets the PDF version in the output file. The behavior is outlined in the following samples:

Example: *Input PDF is version 1.5 and the following command is executed:*

```
pdfoptimize -fv 1.4 input.pdf output.pdf
```

The output file is PDF version 1.5.

Example: *Input PDF is version 1.4 or lower and the following command is executed:*

```
pdfoptimize -fv 1.4 input.pdf output.pdf
```

The output file is PDF version 1.4.

Example: *Input PDF is version 1.3 and the following command is executed:*

```
pdfoptimize -fv 1.4 -fc 8 input.pdf output.pdf
```

If input.pdf contains color images to which JPEG2000 compression is applied, the output file will be version 1.5. Otherwise it will be version 1.4.

-id Set Value in the Document Information Dictionary

Set the value of an info entry key. Examples for keys are "Author", "Subject", "Title", "Producer" or custom attributes.

Example: *Set the title:*

```
pdfoptimize -id Title "My Title" input.pdf output.pdf
```

-isc Enable Conservative Image Processing Strategy

If set, then the different compression types given with -fb, -fc, and -fi are omitted if the current image is neither clipped nor color-converted nor down-sampled. Thus, the original image in the input document is conserved if it has not been altered.

-lf List Fonts

List all fonts and their properties.

Table: List Fonts

Parameter	*Description*	*Example*
FontName	The name of the font. Subsetting-prefixes are not listed as name of the font.	"Arial-BoldMT", "Verdana"
FontType	The font type.	TrueType, Type1

Encoding	The encoding of the font, see examples.	DifferenceEncoding, IntrinsicEncoding, MacRomanEncoding, SymbolEncoding, WinAnsiEncoding
IsCID	Whether the font is a CID font (Character Identifier Font) or not.	CID, Non-CID
IsEmbedded	Whether the font has an embedded font program or not.	Embedded, Non-embedded
IsSubsetted	Whether a font program is subsetted or not. This value is only set for fonts, which have an embedded font program.	Subsetted, Non-Subsetted
Filename	The file name of the font program. This is the name under which the font is saved to file in case the switch -xf is applied. For all non-embedded fonts, there is no file name available (N/A).	fnt12.ttf, fnt2477.cff, N/A

Example: *The following command lists all fonts of a PDF document:*

```
pdfoptimize -lf input.pdf
FontName, FontType, Encoding, IsCID, IsEmbedded, IsSubsetted, Filename
"Arial-BoldMT", TrueType, MacRomanEncoding, Non-CID, Non-embedded, N/A,
"Arial-BlackItalic", TrueType, MacRomanEncoding, Non-CID, Non-embedded, N/A,
"Verdana", TrueType, WinAnsiEncoding, Non-CID, Embedded, Subsetted, fnt38.ttf
```

The first line in the above example is the actual command, the following lines list the output.

See also switch -xf for extracting fonts.

-li List Images

List all images and their properties.

Table: List Images

Parameter	*Description*	*Example*
ObjectNumber	The PDF object number.	9
Width	The width of the image in pixel.	400
Height	The height of the image in pixel.	589
BitsPerComponent	The number of bits that are used to represent one component. This number is in most cases either 1 (bi-tonal) or 8 (RGB, CMYK, Gray).	8
ColorSpace	The color space of the image.	DeviceCMYK, DeviceRGB, DeviceGray, ICCBased,

		Indexed,
Resolution	The resolution in dots per inch (dpi).	96
Filter	The compression filter.	DCTDecode, FlateDecode
ImageSize	The uncompressed image size.	706800
CompressedSize	The compressed image size.	28172
CompressionRatio	The ratio compressed image size divided by uncompressed images size. The smaller this value, the higher the compression.	3.99%
FileName	The file name of the image. This is the name under which the image is saved to file in case the switch -xi is applied.	img9.tif

Example: *The following command lists all images in the file input.pdf. In this case there is one image.*

```
pdfoptimize -li input.pdf

ObjectNumber, Width, Height, BitsPerComponent, ColorSpace, Resolution,
Filter, ImageSize, CompressedSize, CompressionRatio, FileName

9, 400, 589, 8, ICCBased, 96, DCTDecode, 706800, 28172, 3.99%, img9.tif
```

See also switch -xi for extracting images.

-lk Set License Key

Pass a license key to the application at runtime instead of installing it on the system.

-m Merge Embedded Font Programs

Font programs can be merged, if they originate from the same font, e.g. they are of the same type, have the same name and encoding. Merging of Type1 (PostScript) and TrueType fonts is supported.

-ml Set Compression Type for MRC Layers

This option affects only MRC foreground and background layers. The option is followed by a single numerical value indicating the compression type to use for MRC foreground and background layers. For possible values see "Table: Color / Monochrome Compression Types" in the description of -fc. The default is 1 (JPEG compression).

See also Mixed Raster Content (MRC) Optimization for Images.

-mlr Set the Resolution in DPI for MRC Layers

This option affects only MRC foreground and background layers. The option is followed by a numerical value that indicates the target resolution in DPI of MRC layers after down-sampling. The default is 70.

See also Mixed Raster Content (MRC) Optimization for Images.

-mlq Set the Image Quality for MRC Layers

This option affects only MRC foreground and background layers. The option is followed by a numerical value between 0 and 100 to be used as the image quality for MRC foreground and background layers when using a lossy compression for these layers. The default is 10.

See also Mixed Raster Content (MRC) Optimization for Images.

-mm Set Compression Type for the MRC Mask

This option affects only MRC masks. The option is followed by a single numerical value indicating the compression type to use for MRC masks. For possible values see "Table: Bi-tonal Compression Types" in the description of -fb. The default is 6 (CCITT Fax Group 4 compression).

See also Mixed Raster Content (MRC) Optimization for Images.

-mp Set Compression Type for MRC Cut-Out Pictures

This option affects only cut-out images when doing MRC optimization. The option is followed by a single numerical value indicating the compression type to use for MRC cut-out pictures. For possible values see "Table: Color / Monochrome Compression Types" in the description of -fc. The default is 1 (JPEG compression).

See also Mixed Raster Content (MRC) Optimization for Images.

-o Set the Owner Password

The owner password is required to change the security settings of the document. In order to apply permission flags, an owner password must be set. Permission flags are set with the switch **-p**.

Example: *Encrypt a document and set the owner password to "owner".*

```
pdfoptimize -o mypassword input.pdf output.pdf
```

-oc Clip Images

Images in PDF documents can be clipped. This means that only part of the image is visible, whilst the rest is hidden. The switch **-oc** detects these images, reduces their size the area that is actually displayed and replaces the original image by the reduced image. Pre-blended images are not clipped.

Setting **-oc** activates the **-od** option.

-od Optimize Resources

Optimize the resources of the PDF, such as images, color spaces, or fonts. If set, unused resources are removed. Also content streams are re-built.

© PDF Tools AG – Premium PDF Technology

-ol Linearize Only

Do not apply any optimizations, but linearize the file. This can be significantly faster than the option `-ow`. See -ow for more information.

-or Remove Redundant Objects

This option removes redundant objects. I.e. it identifies duplicates of objects and merges them.

-ow Linearize the Output File

Add so called linearization tags to the document. A linearized document has a slightly larger file size than a non-linearized file, and provides the following features (among others):

- When a document is opened through a PDF viewing application plug-in for an Internet browser, the first page can be viewed without downloading the entire PDF file.
- When another page is requested by the user, that page is displayed as quickly as possible and incrementally as data arrives, without downloading the entire PDF file.

Note: In order to make use of a linearized PDF file, the PDF must reside as a 'file' on the web-server. It must not be streamed.

-p Set the Permission Flags

This option sets the permission flags. It is only usable in combination with encrypted documents, i.e. an owner password must be set. By default all permissions are granted. The permissions that can be granted are listed in the table below.

Table: Permission Flags

Parameter	*Description*
`p`	allow printing (low resolution)
`m`	allow changing the document
`c`	allow content copying or extraction
`o`	allow commenting
`f`	allow filling of form fields
`s`	allow content extraction for accessibility
`a`	allow document assembly
`d`	allow high quality printing
`-1` default	allow everything (all permissions are granted)
`0`	allow nothing (no permissions are granted)

© PDF Tools AG – Premium PDF Technology

The parameter 0 cannot be combined with other flags. The parameter –1 is the default, it cannot be set explicitly. In order to combine multiple permissions concatenate them to one string.

Example: *The following command sets the owner password to "owner" and the permission flags to allow "printing in low resolution" and "allow form filling".*

```
pdfoptimize –o owner –p pf input.pdf output.pdf
```

Example: *"High quality printing" requires the standard printing flag to be set too.*

```
pdfoptimize –o owner –p pd input.pdf output.pdf
```

For further information about the permission flags, see PDF Reference Manual section 3.5.2.

-pw Read an Encrypted PDF File

When the input PDF file is encrypted and has a user password set, (the password to open the PDF) the password can be provided as parameter of the switch **-pw**.

Example: *The input PDF document is encrypted with a user password. Either the user or the owner password of the input PDF is "mypassword". The command to process such an encrypted file is:*

```
pdfoptimize -pw mypassword input.pdf output.pdf
```

When a PDF is encrypted with a user password and the password is not provided or is incorrect, the 3-Heights™ PDF Optimization Shell cannot read and process the file. Instead it will generate the following error message:

```
Password wasn't correct.
```

-q Set the Compression Quality

Set the compression quality index for lossy compression methods. This option only applies to JPEG, JPEG2000 and JBIG2 images. A lower value results in a smaller file size but the images are of poorer visual quality. A higher value results in better visual quality, but also a larger file size.

The supported values range from 1 (lowest) to 100 (highest). The default is 75. For images compressions that support lossless compression (JPEG2000 and JBIG), a value of 100 corresponds to lossless compression, any other value represents lossy compression. JBIG2 only supports values that are multiples of 10 (10, 20, … 100).

Example: *The following command sets the quality index to 50. All images types which support the quality parameter are recompressed with this quality index.*

```
pdfoptimize -q 50 input.pdf output.pdf
```

-rs Remove Embedded Standard Fonts

This option removes all embedded standard fonts and replaces them with one of the 14 PDF Standard Fonts. The following font families are removed:

Arial	CourierNewPS	Times	ZapfDingbats
Courier	Helvetica	TimesNewRoman	
CourierNew	Symbol	TimesNewRomanPS	

and their derivatives (they are different for different font families) such as:

Arial,Bold	Arial-Bold	Arial-Italic	ArialMT
Arial,BoldItalic	Arial-BoldItalic	Arial-BoldMT	Courier-Bold
Arial,Italic	Arial-BoldItalicMT	Arial-ItalicMT	Courier-Oblique

A PDF Viewer must be able to display standard fonts correctly, even if they are not embedded. Therefore using this option should not visually alter the PDF when it is displayed. Un-embedding a font decreases the file size.

Un-embedding the font works based on the font's Unicode information. I.e. the un-embedded font's characters are mapped to those of the original font with the same Unicode. Therefore, only fonts with Unicode information will be un-embedded by the 3-Heights™ PDF Optimizer. However, if a font's Unicode information is not correct, un-embedding may lead to visual differences. Whether or not a font's Unicode information is correct can be verified by extracting text that uses the font. Suitable tools for this purpose are for instance the 3-Heights™ PDF Extract Tool or an interactive PDF viewer. If the extracted text is meaningful, the font's Unicode information is correct and un-embedding of the font will not lead to visual differences.

-s Subset Fonts

Embedded fonts can be subsetted. Subsetting refers to only storing those character glyphs of the font that are actually used. Unused character glyphs are removed. The advantage is that the size of an embedded font program (and thereby the entire file size) can be reduced this way (in particular for Asian fonts). The downside is that if text is to be edited, only the characters of the subsetted font can be used.

Strip the File

Remove parts of the PDF file. The following parts of a PDF can be stripped:

-sa Strip article threads.

-sf Strip and flatten form fields and annotations. (This removes interactive features of the PDF.)

-si Strip alternate images (variant representations of the base image)

-sm Strip meta data.

-sp Strip page piece info (private application data).

-ss Strip document structure tree (incl. markup).

-st Strip embedded thumbnails.

-sw Strip spider (web capture) info.

-se Strip everything (all of the above).

-u Set User Password

Set the user password of the document. If a document which has a user password is opened for any purpose (such as viewing, printing, editing), either the user or the owner password must be provided.
Someone who knows the user password is able to open and read the document. Someone who knows the owner password is able to open, read and modify (e.g. change passwords) the document. A PDF document can have none, either, or both passwords.

Example: *Encrypt a document with a user and an owner password.*

```
pdfoptimize -u userpassword -o ownerpassword input.pdf output.pdf
```

-v Verbose Mode

This switch turns on the verbose mode. In the verbose mode, the individual steps performed by the 3-Heights™ PDF Optimization Shell are displayed.

-xf Extract Fonts

Extract embedded fonts and save them to a file. This switch does not extract non-embedded fonts. Be aware that due to copyright reasons, the extract font is not an installable font.

The extracted fonts are stored in the current directory and are named as following:

A TrueType font file is named: `fnt{objno}.ttf`

A Type 1 font file is named: `fnt{objno}.pfb`

A CFF font file is named: `fnt{objno}.cff`

Where {objno} corresponds to the object number of the font in the PDF document. This number can also be retrieved with the option `-lf`.

-xi Extract Images

This switch extracts the images from a PDF document and automatically stores them as TIFF or JPEG.

The images are stored in the current directory and are named as following:

`img{objno}.jpg` for images with JPEG compression, or

`img{objno}.tif` for any other type of image.

Where {objno} corresponds to the object number of the image in the PDF document. This number can also be retrieved with the option `-li`.

6.3 Return Codes

All return codes other than "0" indicate an error in the processing.

Table: Return Codes

Value	*Description*
0	Success
1	PDF Input File could not be opened or invalid parameters
2	PDF Output File could not be created
3	Invalid option or option values were entered
4	PDF Input File is encrypted and password is incorrect or not provided

7 Troubleshooting

7.1 The Output File is Too Large

First and foremost it is important to understand what kind of content there is in the document. There is no point in trying to optimizing fonts when the document contains scanned images only. Document properties, such as embedded fonts and images can be listed using the corresponding listing functions (-li, -lf).

General optimization:

Remove redundant objects and optimize resources using –rs –od can always be set.

For images:

1). Remove redundant objects and strip unnecessary information.

***Example**: Optimize resources and strip all.*

```
pdfoptimize –od –sa –sf –si –sp –ss –st –sw input.pdf output.pdf
```

2). Try setting a lower threshold and a lower dpi for the images.

***Example**: Rescale all images with a dpi greater than 72 dpi to 50 dpi.*

```
pdfoptimize -dt 72 -dr 50 input.pdf output.pdf
```

3). You could also try reducing the quality of the jpeg images with the quality option -q. In many cases using a lossy compression is not significant for viewing:

***Example**: Set the quality index to 60.*

```
pdfoptimize -q 60 input.pdf output.pdf
```

4). Verify what image compression algorithms are applied, the smallest file sizes are usually achieved using JPEG or JPX (=JPEG 2000) for grey-scale and color images and JBIG2 for bi-tonal images. When using JPEG, the quality should be at least 75, when using JPX, it can be set as low as 50.

***Example**:Use JBIG2 and JPX compression.*

```
pdfoptimize -fc 8 –fb 7 –q 50 input.pdf output.pdf
```

For fonts:

5). Apply subsetting to fonts using switch `-s`. This means all glyphs of characters that are unused are removed from the font.

6) Merge font programs using the switch `-m`.

7). Remove non-symbolic embedded fonts. Keep in mind that the appearance when rendering a PDF document with non-embedded non-PDF Standard Fonts is unpredictable.

***Example**: Step 1: List all fonts. This step is optional, but it will give you an overview of embedded fonts, so you know what fonts are embedded.*

```
pdfoptimize -lf input.pdf
```

Step 2: Remove embedded programs for non-symbolic standard fonts and merge

fonts.

```
pdfoptimze -rs -s -m input.pdf output.pdf
```

7.2 The Output File Is Larger Than the Input File

1). The 3-Heights™ PDF Optimization Tool also repairs corrupt documents to a certain extent. This means if relevant data is missing it is recovered. This could possibly lead to a larger file size.

2). If linearization is applied, there is information added to the document. This information contains hints for the browser plug-in, and allows it to specifically download only those objects relevant for displaying a certain page. The linearization information can increase the file size by about 1 to 10%.

7.3 The Selected Compression Type is Not Applied

1) Not all compression types can be applied to all types of images. Check the tables "Table: Bi-tonal Compression Types", "Table: Color / Monochrome Compression Types", and "Table: Indexed Compression Types".

2) The optimization is only applied if it reduces the files size, therefore an image cannot be re-compressed with a new compression that uses more disc space than the original compression. This behavior can, however, be switched off with -ff.

7.4 The Output Document Is Not Encrypted

In order to encrypt the output document, set an owner password using the switch **–o** and permission flags using the switch **–p**.

***Example**: Set the owner password to "mypassword" and do not grant any permissions:*

```
pdfoptimize –o mypassword –p 0 input.pdf output.pdf
```

It is not possible to inherit the owner or user password or the permission flags from the input document.